(51) International Patent Classification 6 :		(11) International Publication Number:	**WO 99/09523**
G06T 15/50	**A1**	(43) International Publication Date:	25 February 1999 (25.02.99)

(21) International Application Number: PCT/GB98/02488

(22) International Filing Date: 19 August 1998 (19.08.98)

(30) Priority Data:
9717656.4   20 August 1997 (20.08.97)   GB

(71) Applicant (for all designated States except US): VIDEOLOGIC LIMITED [GB/GB]; Home Park Estate, Kings Langley, Hertfordshire WD4 8LX (GB).

(72) Inventor; and
(75) Inventor/Applicant (for US only): FENNEY, Simon, James [GB/GB]; 14 Detany Court, St. Albans, Hertfordshire AL1 1TT (GB).

(74) Agent: ROBSON, Aidan, John; Reddie & Grose, 16 Theobalds Road, London WC1X 8PL (GB).

(81) Designated States: CN, JP, US, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

(54) Title: SHADING THREE DIMENSIONAL COMPUTER GRAPHICS IMAGES

(57) Abstract

A three dimensional textured computer graphic image is shaded by firstly providing data which defines the computer graphic image. Textured data is then applied to that image. A set of surface normal vectors corresponding to the texture data are then applied to the image and data defining at least one light source which illuminates the image is also provided. For each pixel in the image a shading value is derived to be applied to that pixel from the set of surface normal vectors and the light source data.

- 1 -

SHADING THREE DIMENSIONAL COMPUTER GRAPHICS IMAGES

This invention relates to the shading of three dimensional
computer graphic images, and especially to graphic images
generated in real time.

5   Many three dimensional computer graphics images are
modelled with perfectly flat or smooth surfaces. Usually
these surfaces are constructed from a plurality of small
triangles to which is applied either flat shading, or
smooth shading as described in "Transactions on Computers"

10   IEEE-20 (6) June 1971 pp 623 to 629 by Gouraud, H.,
graduated shading, or, less frequently Phong shading from
CACM 18(6)June 1975 pp 311 to 317 "Illumination for
Computer Generated Pictures". Visual detail may be
applied to these surfaces via the application of textures.

15   These textures are generally two dimensional images and
the process is similar to having an image painted onto a
perfectly smooth wall. It does not model any surface
roughness or any shading effects which might arise
therefrom.

20   In computer graphics the way in which light interacts with
the surface is referred to as shading. One of the simpler
models used for shading is known as Lambert or diffuse
shading. It is computed as a function of the direction of
the light illuminating the surface and the orientation of

25   that surface. The orientation is represented by a unit
vector perpendicular to the surface (a surface normal).
The light direction is also preferably assumed to be a
unit vector which points from the surface to the point of
illumination. In the case of flat shading the surface

30   normal is considered to be constant across the entire
surface. With Gouraud shading three surface normals
defined at the vertices of each triangle are used. The
shading at the vertices of the triangles is calculated
from these normals. These shading values are then

- 2 -

interpolated across the entire surface. This is a satisfactory approximation in many cases. However, it does lead to shading problems such as mach banding and problems with specular highlights.

5   Phong shading gives a superior result to this because it interpolates the surface normally across the triangle and then recalculates the shading at each pixel. However, both of these per pixel operations are considered to be relatively expensive computationally and, therefore,
10   Gouraud shading is therefore more commonly used.

3D computer graphics often makes use of specular shading in addition to diffuse lighting. Specular shading is the modelling of glossy refections of lights. In both types of shading a common basis for the calculation of the
15   shading to be applied is a vector dot product raised to a power. This is shown in equation 1 below.

$$((1-h)+h.\vec{D}_{light}.\vec{D}_{normal})^p$$

In "simulation of wrinkled surfaces" by Blinn, J.F. in Siggaph 1978 pp 286 to 292 there is proposed the concept
20   of bump mapping. This uses an adaptation of texturing to deviate surfaces normal on a pixel by pixel basis. The texture data used to form the derivation of the normal is referred to as the bump map.
Although the position of the surface is not actually moved
25   in 3D graphic space it appears rough because shading is performed with a surface normal which moves in direction as the surface is traversed.

This process is known as surface normal perturbation.
What is stored in the bump map is an amount by which the
30   surface normal is to deviate from its previous value.
Thus, in order to compute the shading applied to a surface

- 3 -

it is necessary to retrieve data about the deviation of
the surface normal from the bump map prior to applying
this deviation to the surface normal. The surface normal
then has to be renormalised in dependence on the
5   orientation of the surface to which it is applied. The
shading calculation is then performed.

The effect of this leads to realistic dynamic changes in
shading as a light source moves relative to the surface.
However, computationally the scheme is approximately the
10   same as that of Phong shading and so to date has been
restricted to non-real time applications.

We have appreciated that an effect similar to that
proposed by Blinn can be implemented with much less
computational power thus enabling realistic changes of
15   shading to be implemented in real time.

Preferably this is implemented in addition to the usual 3D
computer graphics rendering systems which are in common
usage for texturing and shading.

Preferably, after a surface has been rendered the bump map
20   effects are applied as an additional pass over the
surface. For each image element or pixel a bump map
texture element is obtained in a way identical to the
usual texturing operation. Lighting values are also
interpolated across the surface on a pixel by pixel basis
25   from the light sources in use. The lighting values for a
particular pixel are combined with the bump map texel
(texture element) to produce an alpha value and a colour
and thereby look identical to the usual output of the
texturing engine. These are then supplied to the usual
30   blending units to apply the texture. Unlike the approach
taken by Blinn each texel of the bump map stores the
actual direction of the surface normal after perturbation
rather than the displacements of the surface normal.

- 4 -

These normals are given in the surface's coordinate system which is preferably the polar coordinate system. Lighting values are similarly expressed in terms relative to the surface's coordinate system.

5   The invention is defined with more precision in the appended claims to which reference should now be made.

A preferred embodiment of the invention will now be described in detail by way of example with reference to the accompanying drawings in which:

10   Figure 1 is a block diagram of circuitry a first embodiment of the invention;

Figure 2 is a schematic diagram showing the surface normal and its coordinate system; and

Figure 3 is a block diagram of the bump map hardware of
15   Figure 1.

Figure 4 is a schematic diagram showing the surface normal and a Cartesian coordinate representation system in contrast with the polar coordinates of figure 2;

Figure 5 shows schematically a linear filter applied to
20   texels;

As described above this invention relates to computer 3D graphics rendering systems and is applicable but not restricted to hardware based rendering systems. A hardware based system is described here by way of example.

25   The first embodiment of the invention shown in Figure 1 comprises a modified conventional 3D rendering system. Conventional 3D texture hardware 2 is used to apply

- 5 -

texture to the image and rending hardware 4 then shades
the textured image. Conventionally a single connection is
provided between these two hardware blocks.

In the modified system of Figure 1 a store 6 is used for
surface bump map direction parameters for a number of
different bump maps. This stores a set of surface normals
pointing in different directions in dependence on their
location in the bump map. These are called up by the bump
map hardware 8 which combines the lighting values for a
particular pixel with the bump map data from the store 6
to produce an alpha value and a colour. These are
identical to the usual output of the 3D texture hardware 2
and are then supplied to the usual blending unit which
uses the alpha value to combine the colour with existing
colour at that pixel in proportions dependent on the alpha
value (alpha is between 0 and 1).

Thus, the system applies surface normal perturbation
effects to a surface as one additional single pass to
modify the existing texturing and shading. When it is
determined that for a given surface and picture element
"pixel" that a bump map pass is required, then the
appropriate surface parameters are obtained for that
surface. The surface normal for that pixel is determined
by accessing the bump map texture associated with the
surface in a similar manner to existing texture mapping
methods. A direction parameter is also calculated for the
pixel by interpolation. This is similar to the RGB
interpolation performed for Gouraud shading. Thus the
alpha value and colour value are supplied to the blending
unit.

The bump map surface normals stored in store 6 are encoded
in polar coordinate as shown in Figure 2. Angle S
represents the elevation of the surface normal and goes
from 0 to 90°. Angle R is the rotation of the surface

- 6 -

normal and goes from 0 to 360°. As the surface normal is
a unit vector the length value is always I and so it is
not required to store this. Thus a saving on memory is
achieved.

5   In one embodiment of the invention the per surface
direction parameters for the lighting sources are also
encoded in spherical coordinates with parameters T ranging
from 0 to 90° and Q ranging from 0 to 360°. The dot
product power function of equation 1 would then be
10   implementated as shown below in equation 2.

$$((1-h)+h(\sin(S)\sin(T)+\cos(S)\cos(T)\cos(R\text{-}Q)))^p$$

The parameter H is a weighting value that lies in the range
0 to 1. The surface direction parameters T and Q can be
interpolated in a manner similar to that used in Gouraud
15   shading.

Another embodiment would include the T and H per surface
direction parameters as parameters k_1, k_2, k_3 thus giving the
dot product power function shown below in equation 3.

$$(k_1+k_2\sin(S)+k_3\cos(S)\cos(R\text{-}Q))^p$$

20   Typically these values would be calculated as shown below in
equation 4.

$$k_1=(1\text{-}h); \quad k_2=h\sin(T); \quad k_3=h\cos(T);$$

This gives further flexibility as well as reducing the
complexity of the implementation in hardware.

- 7 -

An embodiment of the invention using the equation shown in
equation 3 is illustrated in Figure 3.

The elevation angle S for the surface normal is first
passed to a sine and cosine unit 10 which computes the
5   sine and cosine of the elevation and applies these to
multipliers 12 and 14 where they are combined with
lighting parameters k_2 and k_3. At the same time, the
rotation angle R of the surface normal has the rotation
angle Q of the lighting value subtracted from it in
10   subtracter 16. The cosine of this angle is then derived
in cosine unit 18. The output of this unit is unsigned
and is fed to a multiplier 20 where it serves to multiply
the output of multiplier 14. The output of multiplier 12
is then passed to an adder 22 where it is added to
15   lighting parameter k_1.

The output of adder 22 and multiplier 20 are then passed
to an add/subtract unit 24. A signed bit 26 supplied by
the cosine unit 18 determines whether the adder adds or
subtracts the output of multiplier 20 from the output of
20   adder 22.

The output of this adder is a signed 11 bit number which
is supplied to a clamping unit which reduces it to the
range 0 to 255 (8 bits) and outputs this to a power unit
30 which raises its value to a power p which is supplied
25   to the power unit.

In this embodiment the S and R values obtained from the
bump map texture are both encoded as 8 bit unsigned
numbers. For S 0 to 255 represents angles of 0 to almost
90° (256 would represent 90° exactly) while for R 0 to 255
30   represents angles of 0 to almost 360° (256 would represent
360° exactly).

- 8 -

The units of Figure 3 show the number of bits and whether
or not those integers are signed or unsigned. U x
represents an unsigned x bit integer. While S x
represents a signed x bit integer.

5   Thus, the alpha output to the blending unit is provided
along with a colour from the existing 3D texture hardware
2. The existing colour and the new colour are then
combined in the blending hardware 4 to produce a new value
for that particular pixel.

10   Using this method has several advantages. Firstly, storage
of surface normals as polar co-ordinates makes the bump
map data compact compared to the method of Blinn which
used surface normal displacements. Furthermore,
renormalisation of the surface normals is not necessary
15   because of the nature of storage as surface normals.
Finally, interpolation of light direction is a relatively
straight forward calculation to be performed since in most
scenes there will only be a small number of light sources
on which the lighting direction has to be based. This
20   enables rendering to be performed in real time.

The bump mapping technique described above has some
shortcomings. These are:

1.   Interpolation of the lighting direction given at each
vertex is 'tricky', as the direction is specified in polar
25   coordinates. Although polar coordinates allow greater
precision with the direction specification and do not need
normalisation, to perform the interpolation requires
significant modification to the iterator units. Because
of this, the hardware can assume that the light direction
30   is constant across each polygon. This effectively
eliminates Phong shading.

- 9 -

2.   For similar reason, bilinear texturing computations
are more complicated. Although some modifications were
made to perform angular bilinear, the actual results are
not ideal.

5   3.   The system cannot model light directions that are
'below' the horizon - these must be converted to an
approximate direction that is on the horizon.

4.   The software interface bears little resemblance to
the actual hardware interface. This means extra work for
10   the drivers or at least to the application.

The second embodiment described below addresses these
issues. To do this there are two major changes to the
implementation:

1.   The light direction vector is now specified in
15   "X,Y,Z" fixed point coordinates. This is very similar to
a typical software interface, in which the light direction
vector is given in floating point coordinates. Ideally,
the floating point vector will have been normalised.

2.   The bump map texel directions are also now specified
20   in Cartesian coordinates, except that one component can be
eliminated due to redundancy. We thus only store "X" and
"Z" per texel.

The idea of specifying bumps and light directions in a
local vertex coordinate system remains the same.
25   Converting a height map to the new format is much easier
than the old, since no trigonometry is required.
Additionally, the new technique includes a 'glossiness'
parameter that allows the modelling of specular
highlights.

- 10 -

As in the first embodiment each texel stores the 'angle' or 'surface normal' of the bumpy surface at that particular texel, and it is assumed that the vector can lie anywhere within a hemisphere, as shown in figure 4.

5   We are not interested in the length of this vector (as it is assumed to be of unit length) but only in its angle. In the first embodiment, this vector was stored using polar coordinates, however these are a nuisance to interpolate.

10   In the second embodiment, the vector is represented in the more usual Cartesian coordinate system. The obvious way to store this would be X,Y,Z, where Y is always positive, and X & Z are signed values, however, we are typically limited to only 16 bits. If, however, we scale the vector

15   such that

$$|x_s| + y_s + |z_s| = 1$$

then there is no need to store the Y component at all, since it can be derived from the other two values. Note that this vector is no longer of unit length. Also all

20   components of this vector are $\langle =1$, and that the length of this scaled vector is also $\langle =1$.

Expressing this in terms of a 16 bit texel, we would have the following:

    UNIT8 TexelX, TexelY;

25   TexelX = ((int) (Xscaled * 127.0f)) + 127;
        TexelZ = ((int) (Zscaled * 127.0f)) = 127;

This packs X and Z as **offset** 8 bit values. That is, a value of 0 represents $^{-127}/_{127}$, while 254 represents $^{+127}/_{127}$. We use this notation rather than the usual 2's complement

30   to make the bilinear interpolation straight-forward.

To extract the X,Y and Z components 'in the hardware', we do...

- 11 -

```
INT9   BumpX, BumpZ;
UINT8  BumpY;


BumpX = (TexelX - 127)* 2;
BumpZ = (TexelZ - 127)* 2;
BumpY = 255 - ABS (BumpX) - ABS(BumpZ);
```

We are guaranteed that Y is positive as
(ABS(BumpX)+ABS(BumpZ)) must be (= 255.   (The above could
probably be expressed better).

TexelX and TexelZ can be the results from the
linear/bilinear/trilinear filtering.

One of the problems with the first embodiment is the
behaviour of the bilinear filtering. With angles, there
is a problem with wrapping around or taking the shortest
interpolation path. This is eliminated with the X/Z
scheme.

The interpolation is performed with just the TexelX and
TexelZ components, and the Y is calculated from the
filtered result. Since these values are in the range
0..255, the standard RGB filtering hardware is directly
applicable. For the following examples, only a linear
'filter' will be used since both bilinear and trilinear
are repeated linears.

Figure 5 shows a view from above looking down on the bump
direction hemisphere. The dotted diamond shape represents
the limits of the scaled X and Z values, which when re-
normalised with the computed Y value would stretch out to
the boundary of the hemisphere. Three example linear
interpolations in X and Z are shown.

For Path A, the interpolation would result in an angle
that goes up and over the pole of the hemisphere - which
is ideal. The previous method would have chosen a path

- 12 -

that ran in a circle 'copying' the circumference. For
Path B, the interpolation would stay close to the
circumference of the hemisphere. *Path C*, should also
result in a sensible interpolation with a rise and fall in
5   the 'Y' component.

The only likely quibble with this scheme is that the rate
of change of the angle may not be constant, but this seems
very minor.

To prevent loss of accuracy with fixed point
10   implementations, it is important that the length of the
vector should not decrease too greatly, since in a fixed
point system, bits will be lost. In this encoding , the
minimum length would occur when $| x_s | = y_s = | z_s | = \frac{1}{3}$,
resulting in a length of $\frac{1}{3}$. This loses less than 2 bits
15   of accuracy, and so is acceptable.

There are two things we must guard against. The first is
that not all possible combinations of 'texel' contents are
valid. Since we have specified that $| x_s | + y_s + | z_s | = 1$
a texel that has $| x_s | + | z_s | \rangle 1$ is clearly invalid. We
20   must therefore, protect against such occurrences.

The second point is that even if the original texels are
valid, there is a small chance that the bilinear unit will
produce X and Z values which also *just* exceed these legal
values.

25   As with the alternate bump format the light direction
vector is stored in Cartesian coordinates. The only space
we have available is the OffsetRGB/original-BumpK values,
as an example we may have 8 bits for each of the X, Y, and
Z components. These values all need to be signed, and to
30   keep accuracy, *it is assumed that the light direction
vector is normalised before conversion to integer*. The
per-vertex values would therefore be calculated from...

- 13 -

```
int8 VertX, VertY, VertZ;

VertLightX = ((int) (LightDir(0) * 127.0f)) & 0xFF;
VertLightY = ((int) (LightDir(1) * 127.0f)) & 0xFF;
VertLightZ = ((int) (LightDir(2) * 127.0f)) & 0xFF;
```

5   Since we are assuming that each vertex light vector is of
       unit length and because we are using 'linear'
       interpolation, the vector for a particular pixel will have
       a length that is ≤ 1. As with the bump map, it is
       important that the in-between vectors are not too short or
10   else too much accuracy will be lost.

       If we assume that the maximum sensible angle difference
       will be 120°, then the shortest vector will be $\sin(30°)$ =
       ½. We will therefore only lose about 1 bit of accuracy due
       to the shortening of vectors.

15   To have the chance of 'smooth' animation, it is important
       that small changes in light direction can be modelled.
       This will be of maximum importance near where the light
       direction = (0,1,0) ie. on the horizon so examining the
       minimum integer variation that seems possible we get
20   [2,254,0]. This appears to be about an angle of 0.1
       degrees, which seems small enough.

       The shading "dot product" computation is much simpler than
       it is with polar coordinates and is implemented in a well
       known manner.

25   To simulate glossy highlights, a 'power' function is
       usually applied to the dot product so that bright areas
       become concentrated. The typical Phong lighting model
       raises the dot product to an arbitrary power, but this is
       too expensive to implement in hardware.

30   A cheaper, but more than satisfactory function is to use a
       quadratic approximation as shown below.

- 14 -

Let X be the result of the dot product,

C be a 'fixed point' 8 bit concentration value, where C=0 (==0.0) gives a linear output, and C=255 (==1.0) gives maximum concentration.

5   We compute...

- $k=C+8$;   *(k is a 9 bit value with 3 bits of fraction)*
- $L=MAX(0, 1023-(k*(1023-X)\gg 3))$;   *L is a 10 bit fractional value*

10   - $Q=(L*L)\gg 10$ *Q is a 10 bit fractional value*
- $P=L+C*(Q-L)\gg 8$;

P is then the fixed point result of the power function. Note that $Q \le L$ and so the final calculation will require signed maths.

15   In total, the highlight function will require 5 add/subtracts and 3 multiplies, although a couple of these are rather simple degenerate cases.

Thus, it will be appreciated that preferred embodiments of the present invention provide a system which enables
20   textured surfaces to be shaded much more efficiently than has been possible.

15

CLAIMS

1.   A method for shading a three dimensional textured
computer graphic image comprising the steps of:
    providing data defining the three dimensional computer
graphic image;
    applying texture data to the image;
    providing a set of surface normal vectors corresponding
to the texture data applied to the image;
    providing data defining at least one light source
illuminating the image; and,
    for each pixel in the image, deriving a shading value to
be applied to that pixel from the set of surface normal
vectors and the light source data.

2.   A method according to claim 1 in which the surface normal
vectors are stored in polar coordinates.

3.   A method according to claim 1 or 2 in which the light
source data is stored in polar coordinates.

4.   A method according to claim 1 in which the step of
deriving a shading value to be applied to a pixel comprises
deriving a colour value and a blending value from the light
source data and combining this colour value with existing
colour data from that pixel in dependence on the blending
value.

5.   A method according to claim 1 in which the surface normal
vector is stored in Cartesian coordinates.

6.   A method according to claim 1 or 5 in which the light
source data is stored in Cartesian coordinates.

7.   A method according to claim 5 in which for each surface
normal only two of the Cartesian coordinates are stored.

8.   A method according to any preceding claim comprising the step of applying a linear filter to the texture data at least once to map values to individual pixels.

9.   A method according to any preceding claim including the step of applying a glossiness parameter to a pixel.

10.   Apparatus for shading a three dimensional textured computer graphic image comprising:
      means for providing data defining the three dimensional computer graphic image;
      means for applying texture data to the image;
      means for providing a set of surface normal vectors corresponding to the texture data applied to the image;
      means for providing data defining at least one light source illuminating the image; and'
      means for deriving a shading value to be applied to each pixel in the image from the set of surface normal vectors and the light source data.

11.   Apparatus according to claim 10 in which the surface normals are stored in polar coordinates.

12.   Apparatus according to claim 10 or 11 in which light source data is stored in polar coordinates.

13.   Apparatus according to claim 10 in which the surface normals are stored in Cartesian coordinates.

14.   Apparatus according to claim 10 or 13 in which the light source data is stored in Cartesian coordinates.

15.   Apparatus according to claim 13 in which for each surface normal only two of the Cartesian coordinates are stored.

16.   Apparatus according to any of claims 10 to 15 comprising means for applying a linear filter at least once to the

texture data to map values onto individual pixels.

17. A method according to claim 10 in which means for
deriving a shading value to be applied to a pixel comprises
means for deriving a colour value and a blending value from
the light source data and means for combining the colour value
with an existing colour value in dependence on the blending
value.

18. A method for shading a three dimensional textured computer
graphic image substantially as herein described.

19. Apparatus for shading a three dimensional textured
computer graphic image substantially as herein described with
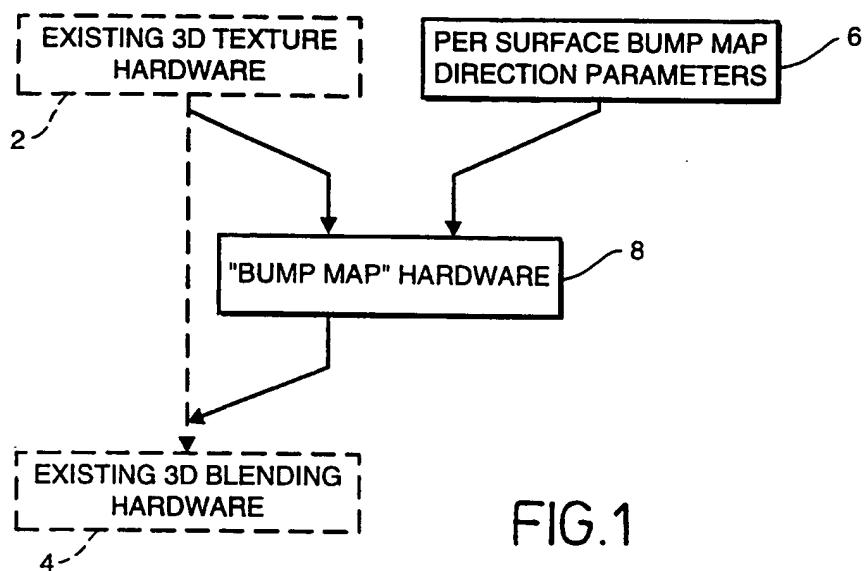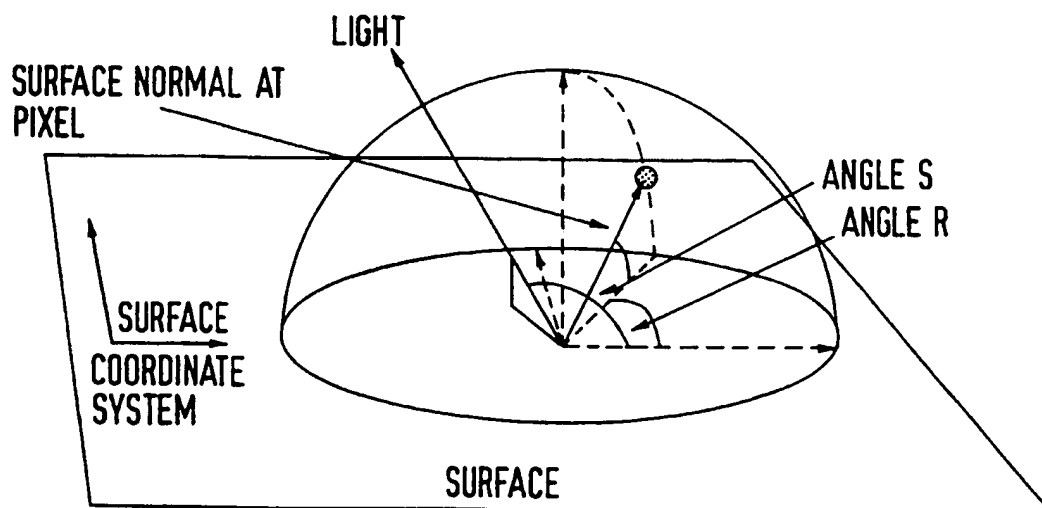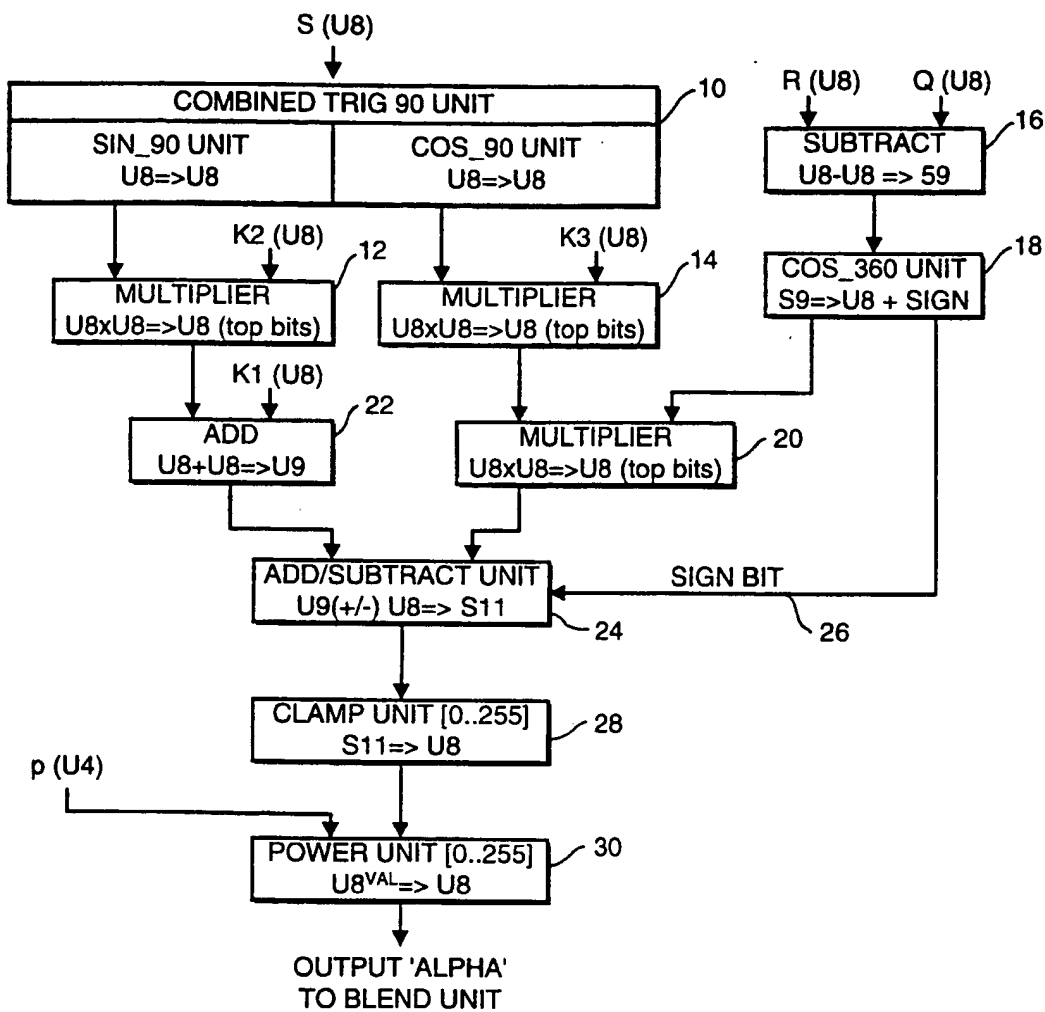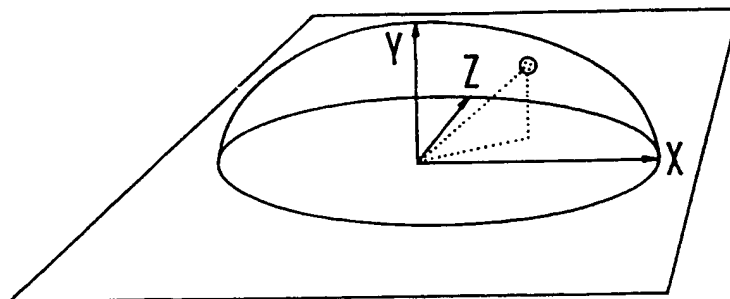reference to the drawings.

FIG.1

FIG.2

S (U8)

| COMBINED TRIG 90 UNIT | | ⟋ 10 |
| SIN_90 UNIT U8=>U8 | COS_90 UNIT U8=>U8 | |

R (U8)   Q (U8)

| SUBTRACT U8-U8 => 59 | ⟋ 16 |

K2 (U8)   K3 (U8)

| MULTIPLIER U8xU8=>U8 (top bits) | ⟋ 12 |

| MULTIPLIER U8xU8=>U8 (top bits) | ⟋ 14 |

| COS_360 UNIT S9=>U8 + SIGN | ⟋ 18 |

K1 (U8)

| ADD U8+U8=>U9 | ⟋ 22 |

| MULTIPLIER U8xU8=>U8 (top bits) | ⟋ 20 |

| ADD/SUBTRACT UNIT U9(+/-) U8=> S11 | ⟋ 24 |

SIGN BIT   ⟍ 26

| CLAMP UNIT [0..255] S11=> U8 | ⟋ 28 |

p (U4)

| POWER UNIT [0..255] U8^VAL=> U8 | ⟋ 30 |

OUTPUT 'ALPHA'
TO BLEND UNIT

FIG.3

FIG.4

FIG.5

INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6   G06T15/50

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6   G06T

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 764 921 A (IKEDO TSUNEO) 26 March 1997 see page 2, line 31 - page 4, line 34	1-4, 8-12, 16-19
X	WO 95 27268 A (ARGONAUT TECHN LTD ;WARNES PETER ROBERT (GB)) 12 October 1995 see the whole document	1,5-7, 10,13-15

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the International search report
4 December 1998	14/12/1998

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-7016	Burgaud, C

Form PCT/ISA/210 (second sheet) (July 1992)

Patent document cited in search report		Publication date	Patent family member(s)		Publication date
EP 0764921	A	26-03-1997	JP	8263694 A	11-10-1996
			JP	8263691 A	11-10-1996
			JP	8263695 A	11-10-1996
			CA	2190938 A	26-09-1996
			WO	9629681 A	26-09-1996
WO 9527268	A	12-10-1995	EP	0753184 A	15-01-1997
			JP	9511085 T	04-11-1997